September 1989 ISSN 0956-9979

VIRUS BULLETIN

THE AUTHORITATIVE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

CONTENTS

EDITORIAL

On the 2nd November of last year, Robert Morris, a 23 year-old graduate student from Cornell university in New York caused havoc when he deposited a 'worm' program onto the Internet computer network in the US. A Federal grand jury has now indicted Morris with gaining unauthorised access to computers and violating the Federal computer crimes statute. If convicted, Morris could receive a five year prison sentence, be fined $250,000 and be ordered to make restitution to persons adversely affected.

At the time of writing, Victoria State police in Melbourne, Australia, have brought their first ever charge of computer trespass against one Deon Barylak, a student accused of infecting a university campus with a PC virus. Two earlier computer viruses, namely Lehigh and Jerusalem (also known as the Hebrew University virus) were discovered on academic campuses. Regardless of whether either of these cases results in successful conviction they do amply demonstrate the connection between virus writing/propagation and educational establishments. This month's *Letter from America* from Jon David, an independent consultant from Tappen, New York, provides an insight into the state of the campus-associated computer virus problem.

Meanwhile, let us all look forward to the start of the new academic term.

TECHNICAL EDITORIAL

Virus Bulletin has not received permission to reproduce this article on CD from the author. Readers can obtain a paper copy of the original issue directly from *VB*.

MACINTOSH CLONES

David Ferbrache

Yet another nVIR B strain has appeared recently for the Mac. Discovered in Minnesota, USA, in August this strain is named nFLU. This brings the number of nVIR B to four, namely AIDS, Hpat, MEV and now nFLU. All of these clones can be produced from nVIR B in the space of a few minutes using a binary editor.

Early virus detection utilities on the Mac relied on recognising resource types, numbers and names added by the virus on infection. Thus disinfectants will search for the nVIR resource type before diagnosing infection. The clones defeat this mechanism, and have forced anti-virus software writers to resort to scanning for byte strings in code resources (such as Virus detective 3.1). A selection of recognition strings is provided in the *Known Macintosh Virus Table* (page 6). Commercial anti-virus products such as SAM which rely on resource detection require regular upgrades to keep pace with new clones or strains. For instance, version 1.0 fails to detect the new nFLU clone.

In addition, the Mac community has realised that there are two different versions of the Peace virus. Ironically, the two principal shareware products recognised different versions of the virus! The variants have different resource names. DR is the latest version, RR the earlier. The DR strain will upgrade any RR infection it detects in the system file.

The Peace virus, having delivered its message of universal peace on March 2nd 1988 is now extinct. The virus was programmed to delete itself after this date. It does, however, show how easy it is to overlook clones of known viruses, particularly when the effects are identical.

LETTER FROM AMERICA

Viruses - The State of the States, of Late

Early in June, I was contacted separately by two local (and unrelated) universities with virus problems. The first had endured virus attacks for more than a month before I was contacted, and was suffering continuing reinfection. I advised the university computer leadership to caution students against taking software off campus, and was told they had already received several calls from irate employers. (I have since heard of more than a dozen companies being hit with the same virus, an Israeli variant, and in each case the virus entry was traced to somebody from this one school.)

The second school reacted differently, calling me immediately upon being hit. Another Israeli variant, this virus treated WP.EXE, the WordPerfect executable, slightly differently (in an apparent attempt to circumvent WP self checking code), and infected programs on 3 1/2 inch disks. Use of infected systems was immediately stopped, a prompt call for assistance made and further infection prevented. Beyond the typical Israeli virus symptoms, both of these viruses caused little boxes to appear (and move) on screens. (The patterns were different at the two universities.)

A third local university, unrelated to the first two, had workstations on one LAN hit with an Israeli variant. Although they unsuccessfully wrestled with the problem themselves for a couple of weeks before seeking help, the virus does not seem to have gone beyond that one LAN.

Two days ago I was called regarding a midwest concern being hit by a LAN virus. (Copies are being sent at this time, so I cannot state it was actually a virus). This (possible) virus reportedly changes files (executable and data) into subdirectories, and further decreases the number of allowable subdirectories. The afflicted company has traced the virus entry (via their administrative usage logs) to a consultant, and the consultant has indicated he has experienced similar manifestations. Coincidentally, the consultant is affiliated with a university.

A southwest concern was hit with what appears to be a standard Israeli virus. They wanted to take formal action, but the security head was going on vacation for two weeks. After the return of the security head, the assistant was away for a couple of weeks, and then there were meetings, backed up work and the like. Over a month has passed and while infections are being cleaned up (?) when they become obvious, this company still has not established virus protection plans ... we all have our priorities. Once again, the initial entry of the virus has been traced to a consultant (since I'm a consultant, you can imagine how much it pains me to say this), and, once again, the consultant is affiliated with a university.

The free access to computers, diskette orientation and concentration of technically astute, if not mature, minds at universities make such institutions much more likely sites for viruses than typical businesses. Employers should be aware of this situation, and establish appropriate procedures for employees with university contact, i.e. new student hires, existing employees taking refresher or advancement courses, anybody taking PC work home with family members that have university contact, etc.

There are several new board-based anti-virus products currently entering the market. These products claim to provide more security than software whilst speeding up certain operations (such as encryption/decryption, viewed as good for virus protection). These boards do not go beyond the functional capabilities of proven anti-virus software, so there is nothing new to look forward to in the functional approach to virus protection. Further, the boards are structured to general PC security products. Now, while this idea of general security added to virus protection may seem worthwhile when you first come in contact with it, the general security is mandatory, not optional, and, unless you happen to fit the security provided, you may find yourself forced to drop the board from consideration.

Although I work virtually exclusively with IBM and compatible PCs I try to remain aware of what is happening in other computing areas. A headline clipped from *MacWEEK* (July 11) contained a cautionary tale. The headline read "Apple dealers get virus," and the story told of Apple, themselves, distributing infected diskettes. Although there have been stories of software vendors providing shrink-wrapped viruses, the idea of one coming from the hardware vendor is rather frightening.

Dr. Jon David

KNOWN IBM PC VIRUSES

Joe Hirst

The following is a list of the **known** viruses affecting IBM PCs and compatibles, including XTs, ATs and PS/2s. The list consists of two parts. The first part of the list gives aliases and brief descriptions, and this also includes a section on reported viruses (which may be completely inaccurate). The second part includes the infective length (the amount by which the length of an infected file has increased), the hexadecimal pattern to use for detecting the virus, and the offset of this pattern within the virus. Viruses referred to in other publications by number almost always refer to the infective length. The hexadecimal pattern can be used to detect the presence of the virus by using the "search" routine of disk utility programs such as *The Norton Utilities*. We hope to publish an article giving a fuller explanation of how to use this table in the near future.

KNOWN APPLE MACINTOSH VIRUSES

David Ferbrache

The following is a list of the **known** viruses affecting Apple Macintosh computers. Each entry includes the name (and aliases) for the virus; a short description of symptoms; together with the characteristic resources or byte sequences which can be used to detect the virus' presence.

Name	Family	Description
nVIR A	nVIR	When an infected application is executed nVIR A infects the system file (adding an INIT 32 resource), thereafter any reboot will cause the virus to become resident in memory, after which any application launched will become infected. There is a delay period before the virus will begin to announce its presence. This announcement is made once every 16 reboots or 8 infected application launches by either beeping or using Macintalk to say "Don't Panic".
nVIR B	nVIR	Similar to nVIR A but does not utilise Macintalk if installed. Beeps once every 8 reboots or 4 application launches.
Hpat	nVIR	Identical to nVIR B but for resource details.
AIDS	nVIR	Identical to nVIR B but for resource details.
MEV#	nVIR	Identical to nVIR B but for resource details.
nFLU	nVIR	Identical to nVIR B but for resource details. Other members of this family are reported to randomly delete files from the system folder.
Peace	Peace	Also known as the Drew or MacMag virus. The virus does not infect applications but only propagates to system files present on hard or floppy disks. The virus was designed to display a message of world peace on March 2nd 1988, and then delete itself from the system file. Two strains of this virus exist, the newer DR strain will replace its predecessor the RR strain on infection.
Scores	Scores	When an infected application is executed Scores will infect the system file, note pad and scrapbook files; the icons for the last two are changed to a generic document icon. In addition two invisible files are created, named Scores and Desktop. Following a boot from the infected system file the virus is loaded into memory. Two days after infection of the system file the virus will begin to infect any application run within 2 to 3 minutes of its launch. After four days any applications run with "ERIC" or "VULT" resources will cause a system bomb (ID = 12) after 25 minutes. After seven days any application with "VULT" resources will find its disk writes returning system errors after 15 minutes of runtime.
INIT 29	INIT 29	When an infected application is run INIT 29 will infect the system file and patch the open resource file trap. Any action which opens the resource fork of a file will cause the fork to be infected. Note that this virus does not require an application to be run for it to be infected. Only infected system files or applications will spread the virus although other files may be infected. This virus will attempt to infect any newly inserted disk causing the message "the disk needs minor repairs" if it is write protected. Sporadic printing problems may also be encountered.
ANTI	ANTI	This is the first virus for the Mac which does not add new resources on infection, the virus instead appends its code to the CODE 1 resource of the infected application. When an infected application is run the virus will install itself in the system heap, and thereafter infect any application which is launched or has its resource fork opened. Unlike other Mac viruses it does not infect the system file, and thus will only become active in memory when an infected application is run. Anti does not spread under multifinder. The virus is also designed to execute automatically a code block on floppy disks which carry a special signature word.
Dukakis	Hypertext	A virus written in hypertalk which when activated will install itself in the home stack displaying the message "greetings from the hyperavenger... dukakis for President ... Peace on Earth and have a nice day". The virus will then propagate to each stack used, displaying its greeting at three week intervals.

Resources added on infection: resource name, number and length in bytes *n* represents the number of the highest allocated code resource:

Virus	System file			Application			Common to both		
nVIR A	INIT	32	366b	CODE	256	372b	nVIR	1	378b
	nVIR	0	2b	nVIR	2	8b	nVIR	6	868b
	nVIR	4	372b	nVIR	3	366b	nVIR	7	1562b
	nVIR	5	8b	-			-		
nVIR B	INIT	32	416b	CODE	256	422b	nVIR	1	428b
	nVIR	0	2b	nVIR	2	8b	nVIR	6	66b
	nVIR	4	422b	nVIR	3	416b	nVIR	7	2106b
	nVIR	5	8b	-			-		
Hpat	INIT	32	416b	CODE	255	422b	Hpat	1	428b
	Hpat	0	2b	Hpat	2	8b	Hpat	6	66b
	Hpat	4	422b	Hpat	3	416b	Hpat	7	2106b
	Hpat	5	8b	-			-		
Scores	INIT	6	772b	CODE	*n*+2	7026b	-		
	INIT	10	1020b	-			-		
	INIT	17	480b	-			-		
	atpl	128	2410b	-			-		
	DATA	400	7026b	-			-		
INIT 29	INIT	29	712b	CODE	*n*+1	712b	-		
Peace (RR)	INIT	6	1832b "RR"	-			-		
Peace (DR)	INIT	6	1908b "DR"						
Anti			-	CODE 1 extended by 1344b			-		

AIDS, MEV# and nFLU have similar resources to nVIR B but named AIDS, MEV# and nFLU respectively.

Characteristic byte sequences: (from Virus detective Ver 3.1)

nVIR resource size < 800b, 2F3A .. 15 bytes .. 00 .. 12 bytes .. 80

INIT29 resource size < 800b, 41FA .. 9 bytes .. 2E .. 7 bytes .. 97

Scores resource size < 8000b, FD38 .. 15 bytes .. BA .. 5 bytes .. A3

Anti last 1344 bytes in in CODE 1, 060CA9 .. 6 bytes .. 43E9

SPECIAL FEATURE

Dr. Keith Jackson

Checksum Methods Used to Detect Virus Attacks

In last month's edition of Virus Bulletin Ross Greenberg discussed the reported appearance of 'killer viruses' in the United States and offered some possible measures to combat them. Here, Keith Jackson provides further guidance about data integrity checking software (Ed).

Many of the products which are marketed as offering protection against attack by a virus operate by detecting when the contents of a particular portion of disk (or memory) have been altered. They calculate a 'checksum' for the data under study, and test this checksum at intervals in the future. If the checksum cannot be recalculated correctly, then the program concludes that the data under test has been altered. The mechanism of alteration cannot be determined by such programs.

A checksum is merely a number calculated by applying a known mathematical algorithm to the data under study. Many products use the term Cyclic Redundancy Check (CRC). This is only one particular type of checksum from many different mathematical possibilities. It is important that the algorithm used for checksum calculation should be cryptographically strong, and cannot easily be circumvented by any other program, malicious or otherwise. I shall explain this point in detail later.

Last month Ross Greenberg mentioned reports of 'Killer Virus' attacks which purport to have been made by a virus capable of actively evading anti-virus programs. Against such attacks, he dismisses protection offered by methods which detect, and/or eradicate, specific viruses. Such methods are always subject to the constraint that they can only detect currently known viruses. New viruses will always evade such protection methods.

There is little doubt that this type of 'killer' or 'second generation' virus is technically possible. Such a virus could succeed in destroying data and/or programs if the proposed methods of protection are either mathematically weak, or poorly implemented.

Ross concludes that "The other alternative, and the only one that will work, is to use a checksum or CRC program ...". I agree with this conclusion in general terms, and would also strongly recommend that any checksum method should use a cryptographically strong algorithm. As he explains, if a weak algorithm is used any alterations can include extra data added with the sole purpose of faking the original checksum when the checksum is recalculated at some future date.

So which algorithm should be used ?

There is an agreed International Standard which is suitable for use by checksum programs (ISO 8731/2). The algorithm described in this standard is commonly known by the acronym MAA (Message Authentication Algorithm). The standard describes how to calculate a checksum for a particular block of data (a message), and how to check that this checksum has not been altered.

Much research work has been done to ensure that the algorithm described in ISO standard 8731/2 is both cryptographically strong, and easy to execute at high speed. I would therefore recommend that any checksum program should implement such an algorithm, or a similarly approved algorithm. Some products already do just this.

Many currently available products use unpublished algorithms. In such cases details of the mathematical operation of the algorithm are kept secret by the author of the software package. This raises the question how can it be shown that the algorithm is any good? There is no simple method to prove cryptographic strength, and you can therefore normally only rely upon the cryptographic competence of the author of the checksum program.

I've already castigated one software package (*Virus Bulletin,* July 89) for using a checksum algorithm that I could reverse engineer within two hours. This was particularly appalling as I make no claims for special cryptographic competence. In simple terms reverse engineering means working out how the algorithm operates from inspection of the checksums calculated for known data files. Authors of such algorithms often have a good reason for keeping details unpublished - the algorithms are weak and insecure.

Be clear about this : If you use a checksum program that uses an unpublished, secret, or proprietary algorithm (all three descriptions are used), you are entirely at the mercy of the cryptographic competence of the author of the software package. Note that this is not related to how well the author can write software, but is related only to his mathematical and cryptographic ability.

Some authors suggest using more than one algorithm for calculating checksums. This is supposed to defeat any direct attack which uses knowledge of the algorithm to replace the original checksum after a change to the data has been made. The exact algorithm in use at any one time cannot be predicted, thus any program relying

upon knowledge of the checksum algorithm is deterred. However having two algorithms only reduces the odds of a successful attack by a factor of two. Three algorithms would reduce the odds of success by a factor of three. This only circumvents the problem rather than solving it, and I believe such an approach should be ignored. Multiple weak algorithms are no real substitute for one cryptographically strong algorithm.

From the user's point of view it is important that any checksum program executes quickly. I have reviewed many checksum programs, and would not use any that I have come across so far on a regular basis for anything more extensive than a couple of files. The reason is very simple. When I switch on a computer I don't want to wait for ten minutes whilst the checksum program does its work, at most I'm prepared to wait a few seconds.

I have yet to come across a checksum program that can check large numbers of files in such a short period of time.

Assuming for now that the aforementioned problems of algorithm strength and speed of execution have been solved, you should then take care that the checksums are stored in such a manner that a malevolent program cannot gain access to them. As all areas of a hard disk are available to all programs on a PC (this is not always true on larger computers), this means that the only secure storage place is on a floppy disk that is removed and stored in a safe place when the checksum program has completed execution. This floppy disk is then required whenever the checksums are recalculated.

Any checksum program that does not allow the calculated checksums to be stored on a disk separate from the disk under study should be avoided. The simple precaution of storing the checksums out of reach of a virus (or any other malevolent program), effectively prevents such a program altering a file, and then altering the checksum to hide this alteration.

This throws up another problem. Who is going to look after this floppy disk ? This is really part of a wider problem which must be addressed at this stage. Who is going to control use of the checksum program ? There are no hard and fast rules for this, but each organisation should lay down clear rules of usage. If this is not done then frankly the use of any security product is a waste of time, and could even be detrimental by inducing a false sense of security.

Any serious checksum program should maintain an audit trail describing how the program has been used.

The use of a checksum program to test a PC's integrity should be regularly monitored by inspection of this audit trail. The audit trail should be stored in encrypted form to prevent inspection and/or alteration by unauthorised personnel. It's content should only be available to authorised security personnel.

In summary, where checksum programs are concerned, as with all things in life, there are compromises to be made. The ideal would be to use a checksum method using a very strong algorithm applied to every file on a hard disk each time that the computer is powered up, and at frequent intervals if the computer is never powered down.

The reality is that difficult decisions have to made on how this ideal is approached.

The algorithm used by the checksum program must be cryptographically strong. If this is not true, then everything else is probably a waste of time. Given this precondition it is essential to store the checksums somewhere they cannot be altered (inadvertently or otherwise), and that the checksum program works at sufficient speed so as not to cause excessive user inconvenience. In simple terms this means that you will have to select which files to protect - testing a complete hard disk is almost always impractical no matter what the manufacturers' literature may say.

Finally forget using checksum methods unless you have a fast PC, otherwise you'll always find things too slow to contemplate.

VIRUS DISSECTION

Joe Hirst

Virus Bulletin has not received permission to reproduce this article on CD from the author. Readers can obtain a paper copy of the original issue directly from *VB*.

TECHNICAL REVIEW

Dr. Keith Jackson

Advanced Systems Protection - ASP

The author of Advanced Systems Protection (Dr. Fred Cohen) can lay some claim to having first identified the concept of a computer virus in a scientific paper published in 1984. I therefore looked forward to trying out ASP.

The ASP documentation states that it is an integrity maintenance system for detecting and eradicating computer viruses and other sources of data corruption. A bold claim indeed. The security features offered by ASP are integrity checking of disk files and critical areas of memory, file encryption, single or multi-user operation, an online help system, LAN operation and an audit trail.

The manual provided with ASP is 38 pages long, but sadly it has no index so you just have to dig around for specific subjects. The table of contents is itself two pages long, so this is a good place to start, but it is no substitute for a real index.

Discussion within the ASP manual provides an intriguing analogy between a computer virus and a missile. Both have a propulsion system, the replication process in the case of the virus. Both use their propulsion system to transport the contents of the missile to a site where it can carry out its preprogrammed task. Although the analogy is rather over the top, there are definite connections between the two parts of the analogy. It certainly prompted a few thoughts.

There are very clear instructions provided on how to commence using ASP. You first boot the computer using the original ASP disk. A checksum is calculated and displayed on the screen. When ASP is executed in the future it will request that this checksum is entered at the keyboard. The checksum remains the same as long as the computer hardware is not changed. Rather confusingly the manual refers to this checksum as a 'Magic Number'. Generation of this 'Magic Number' is a one-off operation only necessary whenever the computer hardware configuration is altered.

The computer is then booted with MS-DOS in the normal manner, and ASP executed. When ASP is running the user is prompted for a sequence of characters known as the 'Integrity Key'. This should be a phrase chosen at random by the user, entered when ASP is first executed, and remembered for future use. It must be entered correctly before ASP will execute. When first executed, ASP also requests entry of the aforementioned 'Magic Number'. There is a delay of about 5 seconds after the entry of the 'Integrity Key' and the 'Magic Number' whilst ASP performs internal testing. A delay of this magnitude is acceptable.

Users can choose whether ASP provides a DOS shell as a stand-alone program, or whether it replaces the usual COMMAND.COM command interpreter. ASP commands can also be executed from the DOS line.

As a DOS shell, ASP provides a list of available files in the current directory, and a set of commands through which MS-DOS is operated. A systems of screens (windows) provides information which replaces the familiar MS-DOS prompt. I have to admit that I found the layout of the screens confusing, and the single character syntax of the various commands difficult to remember.

The help system provides what is probably the most confusing screen of all. Given the function of this screen, this is a grand irony. Each single character command has its function explained with some options separated by a slash, some enclosed in brackets, and some also available as a control character. After much perusing of the manual, and usage of ASP, I have to admit that I'm still not sure of the significance of the different symbols. The manual explains clearly that ASP performs different actions for upper case and lower case input but I found it difficult to correlate the manual with the layout of this single help screen.

75K of memory is required when ASP executes as a stand-alone program (rather conservatively the manual states 80k). This rises to 100K when ASP is memory resident as a replacement for the MS-DOS command interpreter (COMMAND.COM). ASP will work on a floppy system but it really requires a hard disk. The manual acknowledges this point.

The basic principle of ASP operation is that changes should be authorised by users, so that any unauthorised change (e.g. a change introduced by a virus) is noticeable. When you try to execute a new program for the first time, ASP says "No checksum, should I make one ?". The user must reply. After execution of the program is complete, the user is asked whether the file just executed should have been altered. If the reply is yes, then the checksum is recalculated. All this adds to the amount of typing required to execute a MS-DOS program. If a bad checksum is found when the computer is booted, or during ASP execution, then the user is suitably warned.

I don't want to use DOS this way. If I could use ASP invisibly, I might well change my mind.

I'm comfortable with the normal user interface presented by MS-DOS, and with the infrequent exception of using Norton Commander for very complex file manipulation, in the main I don't find DOS shells very useful. To have a DOS shell providing

security features is almost a contradiction in terms. Any extra security should be as invisible as possible, not visible to the detriment of other operations.

If you don't know much about the use of interrupt vectors by MS-DOS, life could get confusing when using ASP with a program that needs to alter an interrupt vector. ASP monitors the interrupt vectors, and requires that any change should only be done using one of the special commands provided with ASP. These commands permit a user to change the interrupt vectors without the overhead of having to recalculate from scratch the integrity checksums maintained by ASP. Most users don't even know what an interrupt vector is, never mind whether or not they wish to change it.

The documentation says nothing about the algorithm used to perform checksum calculations, except that a "strong cryptographic checksum" is used. I therefore cannot comment on its strength (or otherwise). I've said it many times, but using an unpublished algorithm leaves you totally reliant upon the cryptographic skills of the developer. There is no method of measuring the strength of an unpublished algorithm. However, Fred Cohen is a well known and well respected researcher into many aspects of computer security.

A 51K test file can be encrypted (and/or decrypted) by ASP in 3.8 seconds. This corresponds to an encryption rate of 13 Kbytes per second. Such an overhead is present whenever encrypted files are read from disk or written to disk. The caveats outlined above about the algorithm used for checksum calculation apply equally well to the encryption algorithm. They may be one and the same algorithm for all I know, I could not find any explanation in the documentation that says one way or the other.

ASP can be made to maintain an audit trail of executed commands, but as the audit trail is not encrypted there is nothing to stop any malevolent user editing it. An audit trail should always be impervious to user alteration, usually by being encrypted.

I'm not absolutely sure who ASP is aimed at. On a technical level it succeeds with its stated aims, but I don't see it being used by the everyday user. It seems very good at monitoring what it describes as a system's 'integrity', but this assumes that the user is going to persist in using MS-DOS with ASP operational. I for one find ASP far too intrusive, and in all honesty can't be bothered figuring out the arcane syntax used by the ASP commands.

In conclusion, this is definitely not a package for the naive user. Its technical competence is very good, but the level of knowledge assumed is more than trivial. The features offered by ASP are very capable, but don't expect to be able to get the best out of it unless you are an experienced computer user. If it were my product I would throw away the existing user interface, redesign it, and offer an option whereby ASP could operate invisibly.

ASP's documentation states that it makes "integrity protection easy". I agree. However this is at the expense of making MS-DOS even more awkward to use than normal.

Technical Details

Developer: Dr. Fred Cohen, U.S.A. (address not stated).

Vendor (in the UK): PC Security Ltd., The Old Court House, Trinity Road, Marlow, Bucks. SL7 3AN (Tel. 0628 890390)

Availability: IBM PC/XT/AT, PS/2, or any close compatible running MS-DOS or PC-DOS.

Version evaluated: 2:2

Price: £125

Hardware used:

a) ITT XTRA (a PC compatible) with a 4.77MHz 8088 processor, one 3.5 inch (720K) drive, two 5.25 inch (360K) drives, and a 30 Mbyte Western Digital Hardcard, running under MS-DOS v3.30.

b) Compaq SLT/286 (a battery powered laptop portable) with a 12MHz 80286 processor, one 3.5 inch (720K) drive and a 20 Mbyte internal hard disk, running under MS-DOS v3.30.

BOOK REVIEW

Jim Bates

Computer Viruses - Deloitte Haskins & Sells- 62pp priced £4.95

Authors: Eddy Peers and Chris Ennis

This little book makes no pretension of providing technical detail about computer viruses, rather it attempts to provide guidance to non-technical readers about how best to protect systems against the threat from viruses. It is difficult to ascertain which readership the book is aimed at since the computer management aspects are over-simplified while discussions on the strengths and weaknesses of mini and mainframe computers are relatively lengthy. The authors' lack of hard knowledge about computer viruses becomes painfully obvious as you read through the seven chapters. The book's stated objective to "replace speculation with hard fact" fails when a later chapter suggests that "One can imagine a virus on a billing system filling in zeros in bills to be collected; or large amounts in bills to be paid". Either we are dealing with "hard fact" or we are imagining what highly specialised viruses might do. As far as I could see, no virus is mentioned by name and only the vaguest distinction is made between boot sector viruses and parasitic viruses.

The first chapter provides a sketchy description of what a virus is and how it works. The old chestnut about viruses arriving via bulletin boards is mentioned here and quoted repeatedly throughout the book. Distinctions are drawn between Trojans, logic bombs, time bombs and worms and the type of damage each inflicts. There is then a short chapter on the types of machine which may be affected which lists PCs, minis, mainframes and networks. The PC section suggests that among the immediate targets for viruses are the AUTOEXEC.BAT file and the CONFIG.SYS file. It is also suggested that "Another method is to use the spare space at the end of a file, between the end-of-file and the end-of- disk-sector markers, as a home for the virus. Whilst implanting itself it would put in a pointer to the file tail in another program so that the virus will be run at a later date". Within my own knowledge of PC viruses I have never come across even a suggestion of this "method" being used.

The chapter entitled "What happens when you are infected?" provides a guide to the book's overall style. A hypothetical series of events, starting with infection via a "new public domain game" leads eventually to the "manifestation" of the virus. This is described as either an "innocent message" or the cursor going "jay-walking across the screen at random". Further chapters deal in equally vague fashion with the results of infection, preventing infection and recovery from infection.

Final chapters deal with the production of a coherent company policy with regard to computer security. The emphasis seems to be on closing down a system to outside software since this must be where viruses will come from. The possibility of 'respectable' software being infected is not considered, neither is the risk from people using computers at home. Risks attendant upon the actions of disgruntled employees are mentioned although the major defence of a regular backup routine is noted only in passing and with very little emphasis.

It is difficult to produce a book dealing with such a technical subject as computer viruses in a manner that would prove beneficial to non-technical readers. Such a task could only be undertaken effectively by someone with an extensive knowledge of the subject. If the authors have such knowledge, they disguise this fact throughout the book and fail to provide any more information than the average computer manager would pick up from reading the general computer press. No reference is made to training staff in virus prevention and recognition techniques. Mention is made of anti-virus software products, but no discussion of their effectiveness is entered into.

My final impression is that the book is of little use to computer management. In spite of the pre-amble, there are few hard facts about the current virus situation and how it can best be dealt with, although there **is** a lot of speculation about what *might* happen in the future. It is pointed out that the chances of infection by computer viruses are currently very small, but this is apparently just lip-service since the main tone concentrates on the scale of damage possible once an infection is confirmed.

Available from: Publications Department, Deloitte, Haskins & Sells, Melrose House, 42 Dingwall Road, Croydon CR0 2NE

EVENTS

Datapro is holding a one-day seminar on **Logic Bombs, Trojan Horses and Computer Viruses**. It takes place in London on 12 September 1989. Details from Rosemary White at Datapro, UK, Tel 0628 773277.

S&S Consulting Group is holding two one-day 'strategic' seminars on the **Virus Threat**. They take place on 13 September and 16 November 1989 at Rickmansworth, Herts, UK. Details from S&S Enterprises, Tel 0494 791900.

The IBM PC User Group is holding a two-day event on **Security for PCs and Networks**. The event takes place at the Royal Aeronautical Society, London, on 19 and 20 September. Details from Gordon Condrup on Tel 01 863 1191.

Data Security for the Financial Industry, Cafe Royal, London, 12-13 September. European Seminar on **Security in Communications Networks**, London Marriott Hotel, 20 September. Details on both events from IBC Technical Services, Tel 01 236 4080.

A Symposium on **Security and Computer Viruses** takes place at the Wang Institute of Boston University, Corporate Education Centre, Pyngsboro, Massachussetts, USA, from 20-22 September. For details Tel (USA) 508 649 9731.

Sophos Ltd continue a series of **Virus Workshops.** The next available workshops are on 25 September and 21 November 1989 and are held in London and Oxford respectively. Further details from Karen Richardson at Sophos, UK, Tel 0844 292392.

Compsec '89 in conjunction with the EDP Auditors Annual Conference includes a three hour special presentation on the virus threat. The event takes place at the QE II Centre, London, from 11-13 October, 1989. Details from Penny Moon, Elsevier Seminars, UK, Tel 0865 512242.

The Annual Brief on Secure Systems. This annual report on global computer security developments takes place on 28-30 November, 1989 at the Hague, the Netherlands. details from Peter Hoogenboom, The Netherlands, Tel +31 3403 79597.

VIRUS BULLETIN

Subscription price for 1 year (12 issues) including delivery:

US$ for USA (first class airmail) $350, Rest of the World (first class airmail) £195

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, Haddenham, Aylesbury, HP17 8JD, England

Tel (0844) 290396, International Tel (+44) 844 290396
Fax (0844) 291409, International Fax (+44) 844 291409

US subscriptions only:

June Jordan, Virus Bulletin, PO Box 875, 454 Main Street, Ridgefield, CT 06877

Tel 203 431 8720, Fax 203 431 8165